# **Voice Translation Mobile Web App — Technical Documentation**

## **1. Overview**

The **Voice Translation App** is a mobile-first web application that allows users to:

1. Record audio in one language.
2. Transcribe the recorded speech into text using OpenAI's **Whisper (gpt-4o-transcribe)** model.
3. Translate the transcribed text into a target language using OpenAI GPT-4 model via **LangChain**.
4. Convert the translated text into speech using **pygame audio** with OpenAI **text-to-speech (gpt-4o-mini-tts)** model.
5. Play back the audio translation directly in the browser.

The frontend is built in **ReactPy**, a Python-based reactive web framework, and the backend leverages Python services to handle audio capture, transcription, translation, and text-to-speech. The code is hosted on **GitHub** and deployment is planned on **Render**.

## **2. System Architecture**

[User Mobile Browser]

│

▼

[ReactPy Frontend] ──> [Speech-to-Text Service] ──(Whisper/gpt-4o-transcribe)──> Transcript

│

└───────────────> [Translation Service] ──(GPT-4 via LangChain)──> Translation

│

└───────────────> [Text-to-Speech Service] ──(gpt-4o-mini-tts, voice="coral")──> Audio

│

▼

Audio Playback in Browser

### **2.1 Frontend**

* **Framework:** ReactPy
* **Features:**
  + Mobile-first responsive design
  + Language selectors for input/output languages
  + Start/Stop recording button with Enter key support
  + Translate button and Play Audio button
  + Transcript and Translation shown in clean card layouts

### **2.2 Backend Services**

* **Speech-to-Text Service**
  + Records audio using pyaudio
  + Captures audio in chunks in a thread-safe manner
  + Transcribes audio using OpenAI **Whisper** (gpt-4o-transcribe)
* **Translation Service**
  + Uses OpenAI GPT-4 via **LangChain** to translate text from source to target language
* **Text-to-Speech Service**
  + Converts translated text into audio using OpenAI **TTS model (gpt-4o-mini-tts)**
  + Uses pygame or browser audio elements for playback
  + Voice preset: "coral"

## **3. Key Technologies & Libraries**

| **Component** | **Technology / Library** | **Purpose** |
| --- | --- | --- |
| Frontend Framework | ReactPy | Responsive web UI, state management |
| Audio Recording | PyAudio | Capture microphone input |
| Audio Playback | Pygame / HTML <audio> | Play translated audio |
| Speech-to-Text | OpenAI Whisper (gpt-4o-transcribe) | Convert speech → text |
| Translation | OpenAI GPT-4 via LangChain | Translate text between languages |
| Text-to-Speech | OpenAI TTS (gpt-4o-mini-tts) | Convert text → speech |
| Base64 Encoding | Python base64 | Encode audio for browser playback |
| Deployment | Render | Hosting web app |
| Version Control | GitHub | Code repository |

## **4. File Structure (Example)**

voice-translation-app/

│

├─ services/

│ ├─ speech\_to\_text.py # Handles audio recording and transcription

│ ├─ translate.py # Handles translation via GPT-4 / LangChain

│ └─ text\_to\_speech.py # Converts text to audio

│

├─ app.py # Main ReactPy app with UI

├─ requirements.txt # Python dependencies

└─ README.md # Project documentation

## **5. Main Components**

### **5.1 ReactPy App (app.py)**

* Handles all UI logic
* Maintains state for:  
  + source\_lang / target\_lang
  + transcript / translation
  + audio\_url
  + recording flag
* Buttons:  
  + **Start Recording / Stop Recording** → triggers backend recording service
  + **Translate** → triggers backend translation
  + **Play Audio** → triggers backend TTS
* Transcript and translation displayed in **cards**
* Enter key support to stop recording using use\_effect and window.addEventListener("keydown")

### **5.2 Speech-to-Text (speech\_to\_text.py)**

* Uses PyAudio to record microphone input in a separate thread
* Converts audio chunks to **WAV** format in memory
* Calls OpenAI Whisper (gpt-4o-transcribe) for transcription
* Returns transcript string to frontend

### **5.3 Translation (translate.py)**

* Calls OpenAI GPT-4 via **LangChain**
* Inputs: transcript, source\_lang, target\_lang
* Returns translated text

### **5.4 Text-to-Speech (text\_to\_speech.py)**

* Uses OpenAI TTS (gpt-4o-mini-tts)
* Voice option: "coral"
* Returns audio bytes → encoded as **base64** for ReactPy <audio> element

## **6. Deployment**

* Deployment Target: **Render**
* Requirements:  
  + Python 3.11+
  + All dependencies in requirements.txt
* Frontend and backend bundled as **single ReactPy app**
* Audio playback handled entirely client-side

## **7. Usage Flow**

1. User selects **input** and **output** languages.
2. Press **Start Recording** → audio captured via microphone.
3. Press **Stop Recording** (or Enter key) → audio sent to Whisper → transcript returned.
4. Press **Translate** → transcript sent to GPT-4 → translation returned.
5. Press **Play Translated Audio** → translation sent to TTS → audio played in browser.

## **8. Design & UI Notes**

* Mobile-first layout, responsive cards for transcript/translation
* Cool and clean color palette: blues, purples, neutral grays
* Rounded buttons, touch-friendly spacing
* Minimal and uncluttered for readability

## **9. Dependencies**

reactpy

pyaudio

pygame

openai

python-dotenv

langchain

## **10. Future Enhancements**

* Add multi-language speech recognition (simultaneous source detection)
* Option to download translated audio
* Dark mode support
* Animated recording indicator

This documentation covers **architecture, flow, tools, and technical details** for developers or maintainers.